Efficient Authentication and Authorization of Mobile Users Based on Peer-to-Peer Network Mechanisms

Torsten Braun
University of Bern, Neubrückstrasse 10, CH-3012 Bern, braun@iam.unibe.ch

Hahnsang Kim
INRIA, 2004 route des Lucioles, B.P. 93, F-06902 Sophia Antipolis, Hahnsang.Kim@sophia.inria.fr

Abstract

This paper presents a peer-to-peer based authentication and authorization infrastructure to minimize authentication delays when mobile users roam across different wireless networks. The basic idea is to avoid exchanging security information between networks visited by a roaming user and the user's home authentication, authorization, and accounting (AAA) server that is typically located in the home network possibly far away from the visited network. Instead, authentication and authorization of a roaming user shall be supported by an AAA server in the visited network. We propose that the AAA server that is responsible for authentication and authorization in a newly visited network locates the AAA server in the previously visited network and retrieves the required security information from that AAA server. The AAA servers can be organized in a peer-to-peer manner and peer-to-peer mechanisms can be applied for searching and transferring security information between them. We propose several mechanisms for quickly locating the previously responsible AAA server in order to decrease authentication delays. The performance of these mechanisms is evaluated by simulations. Real performance measurements show the rather low performance overhead of application level forwarding used in peer-to-peer networks.

1 Introduction

Efficient authentication, authorization and accounting (AAA, [3], [11]) for roaming users in mobile wireless environments is a demanding challenge. In particular, authentication and authorization need to be performed in real-time in order to provide seamless access to roaming users in wireless networks. Accounting issues are not as time-critical as authentication and authorization and therefore accounting is not investigated by this paper.

Usually, information for verifying the identity of a user is stored at an AAA server in the user's home network (AAAH). The AAAH stores all information about the user such as subscribed services, security information etc. In a typical AAA scenario, a user visiting a foreign network may contact the foreign AAA agent (AAAF) and ask for granting access to network resources (service request). The AAAF is the local AAA entity in the visited foreign network that needs to check whether the user is authorized to access the local network. To validate the service request, the AAAF takes over the role of an AAA client and sends an authentication request to the AAAH. The AAAF is able to identify the AAAH based on the user identification and home realm information provided by the mobile user to the AAAF in the service request. The AAAH has to answer incoming authentication requests and may deliver challenge information back to the AAAF. Then, the AAAF challenges the user and will receive a user authentication response from it. The AAAF forwards the authentication response to the AAAH and the AAAH will evaluate it. In case of a successful authentication, the AAAH will notify the AAAF about that and the AAAF may grant resource access by the user.

An important problem of this procedure is the significant delay, when users are roaming rather far away from the AAAH. The authentication and authorization procedure should be repeated when a user enters a new network and needs to be re-authenticated. The message exchange overhead between visited network (user and AAAF) and the home network (AAAH) may be substantial and the message exchange delay might exceed acceptable delays of real-time applications or even the duration when a user is visiting a network. In the latter case, a user might have already left the visited network before access to it has been granted. For example, typical round trip times measured using ping between Europe and the US west coast over lightly loaded research networks are in the range of 200 ms. Message exchange for authentication and authorization often requires several round trip times.

A solution to this problem might be the introduction of AAA brokers, to which an AAAH can delegate the authentication decision. These AAA brokers are closer to the roaming user and can therefore reduce the delay of the authentication message exchange. However, this requires that the AAA brokers have enough knowledge to perform the authentication and authorization process. Of course, the AAAH should not give symmetric long-term passwords to the AAA broker for authenticating a user, but similar as in cellular networks such as GSM or UMTS, the AAAH can pre-compute authentication data such as [random number (nonce), corresponding authentication result] and deliver these so-called authentication vectors to the AAA broker [1]. Alternatively, short-term keys or one-time passwords can also be used for authentication. In general, we call the security information that needs to be transferred from AAAH to the AAA broker "security context" hereafter. Note that we focus on user authentication but not on device authentication in this paper.

The security context allows an AAA broker to perform a decision on behalf of the AAAH whether a user's request for getting resource access can be permitted or not. Since

the AAA broker owns and controls the security context we call this entity security context controller (SCC) hereafter. Frequently, those security context controllers not only take over AAA broker functions but also might serve as AAAH for users belonging to their own domain. In the following, we therefore assume that SCCs include both AAA brokers and AAAH entities. Security context information is therefore exchanged between SCCs only. If a security context includes authentication vectors or one-time passwords, a SCC must keep track which authentication vectors or one-time passwords have already been used.

The SCC should be selected such that it covers a certain area, where a user is expected to roam. When the user moves to another network, re-authentication can be performed between the user and the (close) SCC. SCCs can be organized hierarchically (cf. Figure 1): The SCCs are interconnected by the network operator and form a tree. SCCs on a lower level cover small areas but are close to the users. SCCs on a higher level control larger areas but are farther away from the users. On the other hand, an SCC on a higher level covers a rather large area and increases the probability that it can serve a roaming user for a rather long time. This avoids the case that new security contexts need to be requested from the AAAH. Previous work [2] has calculated the optimal location of such a SCC in a hierarchically organized network in order to minimize the authentication delay for roaming users.

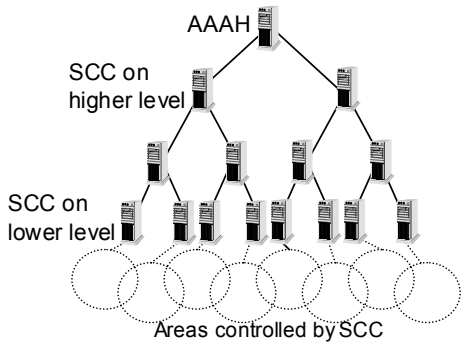

Figure 1: SCC Hierarchy

In [2] it has been assumed that in case a user leaves the area, for which a SCC is responsible, a new SCC must be determined by the AAAH and the security contexts need to be transferred to the newly selected SCC from the AAAH. In this paper, we propose an extension of the concept that allows security context transfer between SCCs without the involvement of the AAAH. This leads to decreasing the authentication delay and in particular avoids the transfer of security contexts from an AAAH that may be far away from the SCCs. It also allows to move the SCCs (AAA broker functionality) even closer to the user, because we can afford to change AAA brokers more frequently due to the fact that security contexts do not need to be retrieved from the (far away) AAAH. In particular, we make use of concepts that have been used in peer-to-peer (P2P) networks. Mechanisms for efficient searches and data replication have been developed by several peer-to-peer networks and those concepts can help to solve the problems addressed above.

In Section 2 we present traditional architectures and procedures for mobile user authentication. Section 3 presents our novel authentication architecture based on a peer-to-peer network established between authentication entities. Section 4 presents performance measurements of application level forwarding as used in peer-to-peer networks and performance evaluations based on a simulation of the authentication architecture. Section 5 concludes the paper and gives some examples for other applications that can take advantage of the P2P search mechanisms discussed in this paper.

2 Authentication and Authorization Architecture for Mobile Networks

Figure 2 shows the message flow for the authentication of a mobile user using SCCs. The service request by the user is received by an AAA client (AAAF), which forwards an authentication request to the next SCC. The SCC requests the security context from the corresponding AAAH and challenges the user with authentication information via the AAAF. The SCC compares the authentication response with an expected response derived from the authentication information and gives the result to the AAAF. The specification of a concrete protocol is beyond the scope of the paper. We rather focus on the general principles for an architecture supporting mobile user authentication. However, we believe that the Diameter protocol [4], which is based on peer-to-peer paradigms, provides a good basis, because it is very flexible and allows being adapted rather easily. Protocol issues of security context transfer have been discussed in [8] and [9].

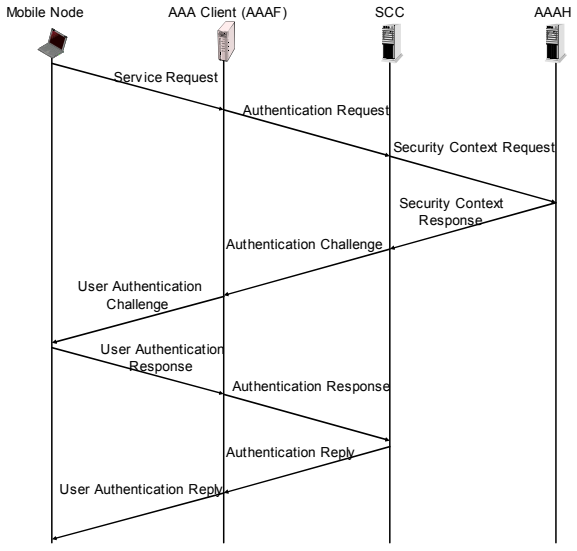

Figure 2 : Authentication Message Exchange

Figure 3 shows the interconnection of the authentication entities. Users connect their end systems to a wireless network and send a service request to the visited wireless network, e.g. wireless network 1. The AAA client that is responsible for wireless network 1 takes the user's service request and sends an authentication request towards the AAAH of the user. The request includes the AAAH as a destination address, but it will be intercepted by SCC 1, which may ask the AAAH to transfer the security context to itself. Note that such a security context transfer is already performed in today's cellular networks between different providers that have established roaming agreements. For

authentication, SCC 1 may challenge the user using a random number (nonce) and compare the response with the pre-computed authentication values stored in its security context. Next, the user may move from wireless network 1 to wireless network 2. Again, the responsible AAA client will receive the service request and forward the authentication request towards the AAAH. SCC 1 still controls the security context for the user and will be able to challenge the user without any interaction with the AAAH.

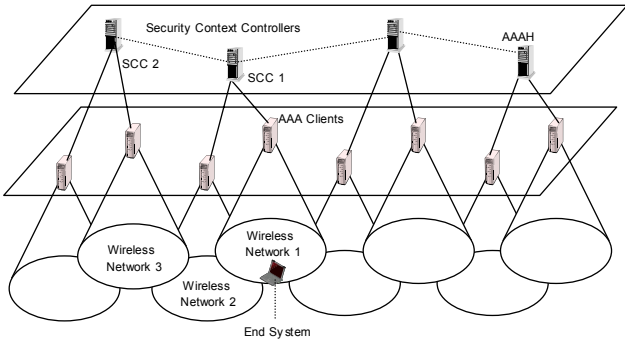

Figure 3: Authentication Architecture

Next, the user moves to wireless network 3. SCC 2 will intercept the authentication request from the AAA client of wireless network 3 and detect that no security context for that user is locally available. To get the security context, SCC 2 has two options: The first (and traditional one) is to request the security context from the user's AAAH. Again, this may add significant delay to the authentication and authorization process. The other option is to search among other SCCs, whether they store a valid security context of the user. For example, SCC 2 may detect that SCC 1 stores such a security context. In that case, the security context can be transferred quickly from SCC 1 to SCC 2 and the authentication process can proceed without contacting the AAAH.

3 Peer-to-Peer Network Technology for Security Context Transfer

3.1 Motivation

To support fast security context transfer we propose to make use of peer-to-peer mechanisms for several reasons:

- P2P networks have been invented in order to efficiently search resources such as audio files. Instead of exchanging audio files, we propose to use P2P mechanisms for locating and transferring security contexts between SCCs. As in other P2P networks, the peer nodes store (key, value) pairs. In our case, the key is a unique identifier for a user and its security context. The value is the current node storing the security context for this user.
- P2P networks support replication and caching. The transfer of security contexts from an AAAH to an SCC can be considered as creating a replicate of the user's security context at the SCC. One has to make sure that authentication vectors are not used multiple times but only once for authentication. Security contexts for a single user with different valid authentication vectors can exist at various SCCs simultaneously.
- P2P networks are able to organize themselves and adapt to changing network conditions. This allows that SCCs discover each other and set up a robust network in order to exchange authentication messages. Such a network should also tolerate node failures and to allow adding new nodes dynamically.
- P2P networks can be used to realize closed user groups. In particular, the set of SCCs need to communicate in a secure manner preferably using strong authentication and encryption mechanisms for security context transfer.

3.2 Peer-to-Peer Based Authentication Architecture

We propose to organize the SCCs in a peer-to-peer network. SCCs could detect each other using P2P mechanisms such as limited broadcast searches or via bootstrap nodes as required for Gnutella [7]. The result of the detection phase should be a mesh of SCC nodes with P2P links between the nodes. Preferably, nodes that are within the same administrative domain or sub-domain and that are geographically close establish links to each other. We also assume that the SCCs can establish secure links to each other based on standard authentication and encryption mechanisms such as IP Security [10]. The SCCs build some kind of secure P2P network and can be assumed to trust each other as it is the case in today's cellular networks.

Each node might be responsible for managing and storing the security contexts of a set of nodes assigned to it. In this case, it acts as an AAAH, e.g. the node indicated by a circle in Figure 4 might be the AAAH for the roaming user represented by the mobile end system.

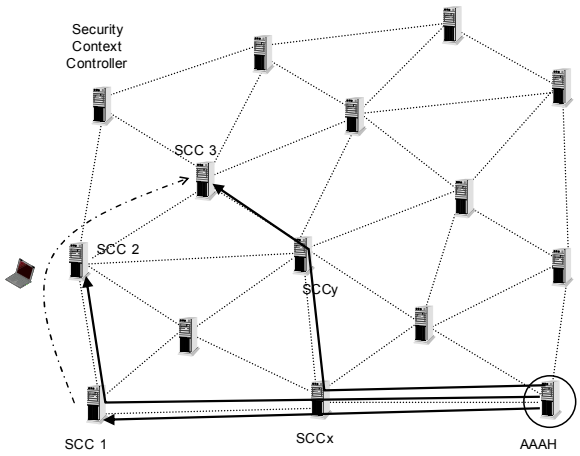

Figure 4 : P2P Organization of Security Context Controllers

In Figure 4, we assume that our user is at first in the area managed by SCC1. SCC1 will request the security context from AAAH. During the security context transfer from AAAH via SCCx to SCC1, pointers to the current security context held at SCC1 will be stored along the forwarding path, i.e. at AAAH and SCCx. After receiving the security

context, SCC1 will broadcast the presence of the security context to its neighbours. This broadcast message should contain the following information: user_ID, timestamp, SCC_ID, TTL. In order to limit the broadcast traffic we propose to limit the broadcast range, e.g. to two hops. Limiting the broadcast range can easily be achieved by the TTL (Time To Live) value. Each forwarding hop needs to decrement the TTL value and is not allowed to forward messages with TTL=0. The timestamp can be used to detect multiple receptions of a single message. Received messages should be stored in a cache with a lifetime that is sufficiently large to detect duplicated messages, e.g. a few seconds. Broadcasting the presence of security contexts should be repeated after a certain time interval (broadcast interval). Simultaneously, broadcast receivers should delete received broadcasts after another interval (broadcast expiration interval) that is a multiple of the broadcast interval in order to tolerate broadcast message loss. This mechanism ensures that only current pointer information (a pointer to the SCC that has been used most recently by the user) is kept at the neighbour nodes and it avoids that outdated information is stored at some SCC. SCC1 should also periodically inform AAAH about that it is still controlling the user's security context (update interval). This update refreshes the pointer information along the path between the current SCC (SCC1) and AAAH, e.g. at SCCx. Since it might happen that multiple SCCs are controlling a security context of a particular user, a timestamp with the last authentication time for the user should be added. If a SCC along the path towards the AAAH receives updates from different SCCs, only the update with the most recent authentication time should be forwarded to the AAAH. Again, the pointer information expires at the nodes along this path after an interval that is a multiple of the update interval (update expiration interval). When the user now moves to an area controlled by SCC2, SCC2 should already know that SCC1 was the previous SCC controlling the security context. Instead of requesting the security context from AAAH, the security context can be requested from SCC1. Only in the case that the security context can not be used any more for authentication, e.g. if all authentication vectors have been used, the new SCC should request a new security context from the AAAH.

In our architecture, the security context does not need to be transferred completely from the previous SCC to the new SCC. The previous SCC might send only a part of it to the new SCC and keep some authentication vectors. Only the SCC possessing an authentication vector is allowed to use it for authentication. By transferring an authentication vector to another SCC, the sending SCC forwards the right to use the authentication vector to the receiving SCC. Transferring only a part of the authentication vectors might be helpful for situations where the previous SCC might be contacted again by the respective user after the security context transfer has been completed. This might happen if the user moves back again to the area controlled by the previous SCC. Another reason might be that some pointer information to the current SCC has not been updated properly. In that case, there might be some pointer information still pointing to the previous SCC but not to the new SCC. In this case, we can avoid redirection and support authentication by those kept authentication vectors. Therefore, SCC1 should keep these authentication vectors and store that it has transferred the security context to SCCs for a longer time interval (larger than the update expiration interval and the broadcast expiration interval), because it might happen that other nodes do not become notified about the security context transfer to SCC2. Then, these nodes might answer a request message with a pointer to SCC1. SCC1 should in that case either use his stored and unused authentication vectors or redirect to SCC2.

After the security context transfer, SCC2 informs the AAAH that it is now controlling the user's security context. If the information travels along the path SCC1 – AAAH, all other pointer information to this security context is updated. For example, SCCx replaces the pointer information to the user's security context and points to SCC2 instead of SCC1. The information might alternatively travel from SCC2 via SCCy (but not via SCCx) to the AAAH. In that case, SCCx might still include some pointer information to SCC1. If it should happen that due to that pointer information another security context transfer request reaches SCC1, it still can support such a request and transfer some unused authentication vectors that have been kept before and that have not been transferred to SCC2. Also the user might travel back to an area controlled by SCC1 after some time. In that case, the kept and unused authentication vectors can be used to support a quick authentication without security context transfer from SCC2 to SCC1.

It may also happen that the user moves to an area with an SCC that did not receive a broadcast message from the previously responsible SCC. This might happen if the user switched off his end system after leaving the previous network and switches it on in a network that is far away from the previous one. In that case, the responsible SCCs are far away from each other and do not receive broadcasts from each other. The same happens if the user stays within the same geographical area but moves to another network provider. For example, the user might first be connected to a WLAN, but might then move out of the WLAN range and connect to a cellular network. This will result in a network provider change and possibly the newly responsible SCC is not in the close neighbourhood of the previous SCC. It might also be the case that the user moves very quickly to an area that is out of the broadcast range.

If the broadcast mechanism is not successful, the new SCC does not know the previous SCC. In such a case, it has to forward a security context request towards the AAAH. If the request passes a node with some pointer information, that node might return the pointer information to the requesting SCC. For example, we assume in Figure 4 that our user disconnects from SCC1, switches its device off, moves to SCC3, and re-connects to the new network. We also assume that broadcasts are only sent to direct neighbours. In this case, SCC3 does not know the previous SCC and forwards the request via SCCy towards AAAH. At SCCx the request meets security context pointer information describing that SCC1 is the current SCC. SCCx returns this information to SCC3. SCC3 contacts SCC1 in order to retrieve the security context from SCC1 and becomes the newly controlling SCC of the user. It should then also notify AAAH about the security context transfer.

By analysing this notification message, SCCy and SCCx will then have pointer information for the user's security context. The pointer points to SCC3 then.

Note that the mechanism described in this section only makes sense, if the AAAH is far away from the previous and the new SCC as well as the SCC with the pointer information (SCCx in the example above) and if these three SCCs are rather close to each other. Otherwise, it would be more efficient to directly request a new security context from AAAH. The SCC with pointer information should estimate and decide which of these two alternatives is better. This may be performed based on hop count information. A simple decision could be based on the evaluation of the estimated distances between the different nodes. If the distance of the deciding node to the AAAH is larger than the sum of the two distances from the deciding node to the new and previous SCC respectively, the node should decide to redirect the service context request to the previous SCC.

The proposed mechanism is very similar to mechanisms proposed in peer-to-peer networks. In such systems key-value pairs are stored at those nodes with IDs that are resulting by applying a hash function to the key. Each key has a root node and that root node may be responsible for storing a certain set of keys. One example is the Oceanstore [6] peer-to-peer file system. Each file has a unique ID and that ID is mapped to the node ID of the file's root node. The root node then holds an entry pointing to the node storing that file and nodes requesting the file may easily contact the root node in order to learn which node is storing the file by applying the hash function to the unique file ID.

4 Performance Evaluation

4.1 Application Level Forwarding Performance

In our investigations we assumed that application level message forwarding between SCCs does not add significant delay compared to IP level forwarding, if both application and network level forwarding use approximately the same paths. In this section we investigate the impact of application level message forwarding compared to IP level forwarding and the experiments discussed below will confirm our assumption. Propagation delays will more and more dominate communication delays in the future while the processing of messages will take less and less time with increasing processing power in intermediate and end systems.

For our measurements we used ten Linux PCs in a common LAN at INRIA Sophia Antipolis (France) and one Linux PC located at University of Bern (Switzerland). Both organizations are connected to their national research and education networks (RENATER and SWITCH), which are interconnected via the multi-gigabit pan-European data communications network GÉANT. Figure 5 shows the message round trip times of the performed experiments. In the first experiment (2 local hosts), TCP messages have been exchanged between two hosts of the same LAN (0.2 ms). Forwarding TCP messages between two hosts via eight intermediate hosts (10 local hosts) increases the round trip time to 3.6 ms. The round trip time on ICMP level (ping) between one host at Sophia Antipolis and one host at Bern via twelve routers in between (1 local, 1 far host (ping)) increases the delay to 30.2 ms. The round trip time on application level between the two hosts (1 local, 1 far host) is the same. In the last experiment, the TCP messages have been first transmitted from a host at Sophia Antipolis via eight hosts at Sophia Antipolis, before the message is transmitted to the host at Bern (9 local hosts, 1 far host). The response is returned along the reverse path. The delay of eight intermediate hosts adds very little delay (< 4 ms) compared to the IP level forwarding delay of approximately 30 ms. The results show that application level forwarding overhead is very low and that the delay added by the network is dominant.

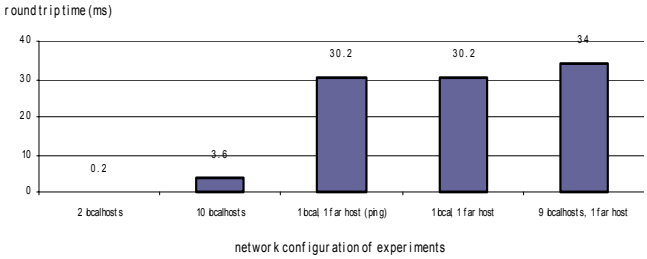

Figure 5: Application Level Forwarding Delay

4.2 Performance of P2P Based Authentication

For the evaluation of the P2P based authentication mechanism, we now assume to have a large grid (1000 x 1000) of 1 million SCC nodes. Each node has coordinates (x, y) with x, y ∈ [0…999] indicating its location in the grid. Each node has four neighbours and we assume that node (500, 500) is the AAAH for the user's service context. This structure is very similar to the CAN [5] peer-to-peer network, where each node has also four direct neighbours.

In case a mobile user changes the network and the broadcast mechanism does not help to resolve the previous SCC, we have to search for the previous SCC by transmitting a security context request message towards the AAAH (see Figure 6). If the request meets on its path towards the AAAH a node knowing the previous SCC, it can return an answer to the new SCC. Otherwise the request arrives at the AAAH, the AAAH transfers a new security context to the new SCC, and the old security context will automatically expire.

In the following evaluation we assume that the costs for retrieving the security context from the AAAH are equal to 2 * N, with N = number of hops between new SCC and AAAH. The costs for retrieving the security context from the previous SCC are equal to 2 * (N' + d) with N' = number of hops between the new SCC and a node with pointer information to find the previous SCC and d = the number of hops between the old and the new SCC.

In the evaluation we selected arbitrary pairs of SCC nodes, i.e. a new SCC and a previous SCC. The path from the SCC nodes to the AAAH is selected according to three different forwarding strategies:

a) Adapt x coordinate first (x first)

b) Random forwarding (random)

c) Anchor based random forwarding (anchor)

With all strategies message forwarding makes always progress towards the AAAH (see Figure 6). With the ***x first*** approach, the message is forwarded such that a node with the same x coordinate as the destination is reached as fast as possible. This mechanism should allow that the search message finds some pointer information at the nodes with the same x coordinates as the AAAH. The forwarding decision is done in a deterministic way. However, if the new and old SCC differ in the y coordinate and differ both significantly from the x coordinate of the AAAH, it takes rather long until a search message can meet some pointer information.

Random forwarding makes random decisions whether to make progress in x or in y direction. A new SCC having completely different y coordinates than the previous SCC might quickly find the pointer information set along the path from the previous SCC and the AAAH. On the other hand, two nodes close to each other may establish two completely different paths to the AAAH.

Anchor based random forwarding visits always some anchor nodes. Anchor nodes might be nodes with special coordinates, e.g. x/y coordinates which are multiples of 5 as depicted in Figure 6. The path from a SCC to the AAAH should always visit one of the next anchor nodes towards AAAH. The path between anchor points is random. If a message has reached an anchor node, there are up to three candidates for selecting the next anchor point. Also this selection is random. For our evaluation we used two levels of anchor points: The lower level includes anchor points with x and y coordinates that can be divided by 10. The higher level includes anchor points with x and y coordinates that can be divided by 100. A message is always forwarded to the next higher level anchor point. On the path towards that higher level anchor point, low level anchor points must be visited.

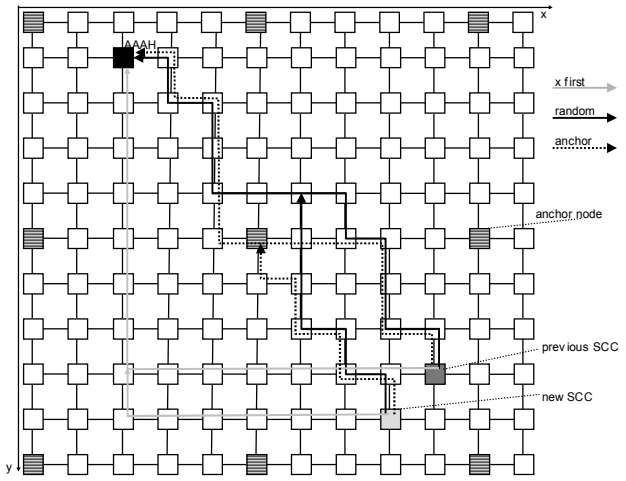

Figure 6: Grid-like organization of SCC nodes

In Figure 6, the top left node has coordinates (0, 0), the previous SCC has coordinates (9, 8) and the new SCC has coordinates (8, 9). The two SCCs are two hops away from each other. (2, 1) are coordinates of the AAAH. The distance from the SCCs to the AAAH is therefore 9 – 2 + 8 – 1 = 8 – 2 + 9 – 1 = 14 respectively. Using random forwarding, at node (6, 4) the security context request from the new SCC meets a node with pointer information for the user's security context. The distance between this node and the new SCC is 8 - 6 + 9 – 4 = 7. The costs for retrieving a security context from AAAH are 2 * 14 = 28. The costs for retrieving the security context from the previous SCC are 2 * 7 + 2 * 2 = 18. Retrieving the security context from the previous SCC is 35 % less costly than retrieving the security context from AAAH.

The number of nodes that must be traversed before meeting a node with a pointer to the security context information depends on the distance between new and previous SCC. Figure 7 shows the relative costs for retrieving security context information for random SCC pairs by applying the three algorithms mentioned above in comparison to retrieving the security context from AAAH. The relative costs can be calculated by (N' + d) / N. The distance between previous and new SCC is given by the number of hops. For all simulations, we have chosen the AAAH in the middle of a 1000 x 1000 grid of nodes, i.e. at coordinates (500, 500). For small distances between previous and new SCC, the algorithms with random forwarding perform better than the deterministic algorithm ("x first"). We also see that the anchor based random forwarding performs always better than any other algorithm even for large distances such as 100 hops between old and new SCC. The purely random based mechanism performs well for small distances, but for large distance values this algorithm performs worse than the deterministic one. One should take into mind that in the case of a roaming user, the probability that two SCCs are far away from each other is rather low, since those SCCs usually cover very large geographic areas. Also different networks operated by different providers in the same country are probably not too far away from each other in the peer-to-peer network. Even if two SCCs are less than 10 hops away from each other, a performance gain of more than 4 can be achieved compared to the traditional case when the security context is retrieved from the AAAH. For large distances, the mechanism does not perform worse than the traditional one.

A further improvement of the concept is the instantiation of several SCCs that are responsible for the security context of a particular user. In this case, the authentication vectors might be distributed over these multiple SCCs. The probability that a security context transfer request meets a node with pointer information to one of these SCCs increases with the number of SCCs. In our evaluation we put 1, 10, 20, 40, and 80 SCCs that all have the same distance to the new SCC and that have a security context for the roaming user. Figure 8 shows significant performance gains by distributing a security context to a rather low number of SCCs, but we see that distributing the security contexts to more SCCs has certain limitations.

Finally, we analyse the required cache memory in the SCCs to support our mechanism. If we again assume a grid of 1000 * 1000 nodes, the average number of intermediate nodes between any 2 nodes of the grid is 667. The number 667 can be calculated by selecting any possible combination of node pairs in a 1000 * 1000 grid and

calculating the average distance over all combinations. Assuming randomly roaming users, 667 nodes (from 1 million nodes) have to store pointers to a user's current SCC. Assuming 10^9 users, user IDs and SCC addresses of 128 bits length each, this requires 2 * 128 bits * 667 * 10^9 = 19.4 TB memory in total and 19.4 MB memory at each SCC in average.

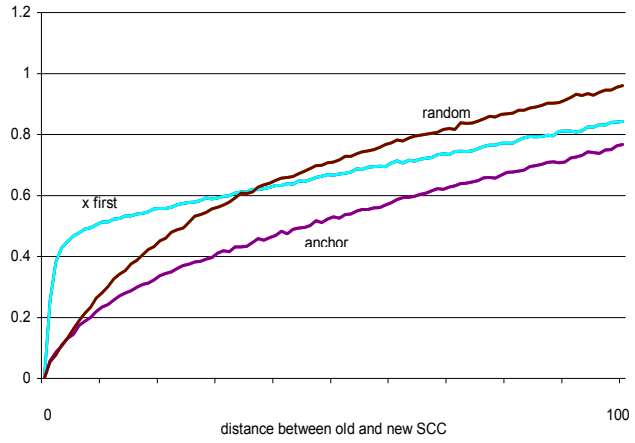

Figure 7: Relative Costs of Security Context Transfer based on P2P techniques compared to traditional approach

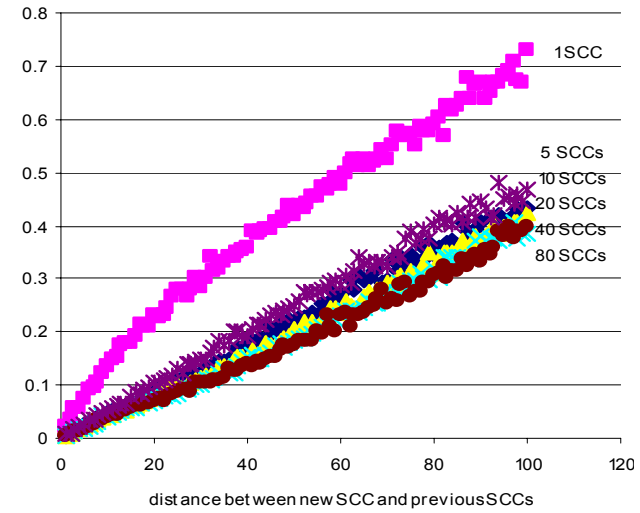

Figure 8: Relative Costs of Security Context Transfer with several previous SCCs and using anchor forwarding compared to traditional approach

5 Conclusions and Outlook

We have presented an architecture for mobile user authentication and authorization based on a peer-to-peer organization of AAA entities. The architecture is based on security context transfer between the previous and the new security context controller. We have analysed different algorithms to find the previous security context controller efficiently in order to minimize the delay for the authentication process. An algorithm based on anchor points and randomly choosing those anchor points as well as randomly walking between the anchor points has achieved the best performance.

The three presented algorithms, and in particular, the anchor based random forwarding algorithm can also be used to solve other problems than presented in this paper.

- One potential application is mobility management by a peer-to-peer network. A mobile end system might connect to a foreign network and to close peers responsible for this foreign network. Then it transmits its new location via the intermediate peers towards a root peer that keeps track of its location. Other peers that desire to determine the mobile node's position also transmit search request messages towards the root peer and might meet a peer along the path that already knows its position.
- Another application is the organization of source-specific multicast trees for P2P based multicast. In this case, new group members need to send join messages towards the multicast source. In order to join the multicast tree, it might be sufficient if the join message meets an already existing branch of the tree. We expect that the search mechanisms based on random decisions will meet the multicast tree earlier.

6 References

[1] H. Kim, H. Afifi: Improving Mobile Authentication with New AAA Protocols, IEEE International Conference on Communications (ICC) 2003, Anchorage, USA, May 2003

[2] H. Kim, W. Ben-Ameur, H. Afifi: Toward Efficient Mobile Authentication in Wireless Inter-Domain, 3rd Workshop on Applications and Services in Wireless Networks (ASWN), Bern, Switzerland, July 2003

[3] C. Rensing, Hasan, M. Karsten, B. Stiller: AAA: A Survey and a Policy-Based Architecture and Framework, IEEE Network, November/December 2002, pp. 22-27

[4] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, J. Arkko: Diameter Base Protocol, RFC 3588, September 2003

[5] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker: A Scalable Content Addressable Network, ACM SIGCOMM 2001

[6] S. Rhea, C. Wells, P. Eaton, D. Geels, B. Zhao, H. Weatherspoon, J. Kubiatowicz: Maintenance-Free Global Data Storage, IEEE Internet Computing, September/October 2001, pp. 40-49

[7] M. Ripeanu, A. Iamnitchi, P. Foster: Mapping the Gnutella network, IEEE Internet Computing, Volume: 6 , Issue: 1 , Jan.-Feb. 2002, pp. 50 – 57

[8] M. Georgiades, N. Akhtar, C. Ploitis, R. Tafaziolli: AAA Context Transfer for Seamless and Secure Multimedia Services over All-IP Infrastructures, 5th European Wireless Conference (EW'04), Barcelona, February 24-27, 2004

[9] H. Wang, A. Prasad: Security Context Transfer in Vertical Handover, 14th IEEE 2003 International Symposium on Personal, Indoor, and Mobile Radio Communication Processing, Beijing, September 7-10, 2003

[10] William Stallings: IP Security, Internet Protocol Journal, Vol. 3, No. 1, March 2000, pp. 11-26

[11] D. Mitton, M. St.Johns, S. Barkley, D. Nelson, B. Patil, M. Stevens, B. Wolff: Authentication, Authorization, and Accounting: Protocol, Evaluation, RFC 3127, June 2001